第3章 最新のフラッシュFPGAを動かしてみた

――セキュリティ機能に期待，デバイス乗り換えがある場合は設計入力に注意

浅田朋範

ここでは，米国Actel社のフラッシュFPGA「ProASIC3/3E」の評価事例を紹介する．ふだんSRAM FPGAを使用している筆者が，今回初めてフラッシュFPGAを動作させた．開発環境，とくにテストベンチ生成ツールの使い勝手は良好だった．また，フラッシュFPGAならではのセキュリティ機能などに注目しているという．なお，本記事で使用するFPGA設計ツール「Libero IDE 6.2」の評価版は，本誌付属のCD-ROMに収録されている．（編集部）

米国Actel社というと「アンチヒューズFPGAのベンダ」というイメージが強いのですが，近年はフラッシュFPGAも出荷しており，製品ラインナップが広がっています．ただし，今のところフラッシュFPGAは，それほどポピュラなデバイスとは言えません．SRAM FPGAと比べた場合の優位点やツールの使い勝手など，設計環境面について知られていないことは多く，実際に製品に適用するとなると，事前に確認しておかなければならない項目が多々あります．

筆者はふだんSRAM FPGAを使用しており，フラッシュFPGAを評価するのは今回が初めてです．本稿では，おもにSRAM FPGAユーザの視点で同社の最新デバイスである「ProASIC3/3E」の特徴を説明していきます．続いて，同社のFPGA設計ツールである「Libero IDE 6.2」の使用方法を，設計フローに沿って解説します．

1 SRAM FPGA vs. フラッシュFPGA

ProASIC3/3Eのダイを**写真1**に，シリーズの概要を**表1**に示します．以下ではSRAM FPGAと比べながらフラッシュFPGAの特徴を説明していきます．

● コンフィグレーションROMが不要に

ProASIC3/3Eでは論理領域や結線用のプログラム素子が不揮発性のフラッシュ・メモリで構成されており，電源をOFFにしてもデバイスの論理情報は保持されます．SRAM

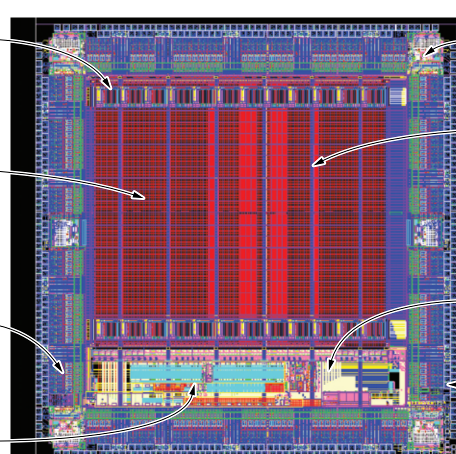


写真1
ProASIC3/3Eのダイ
SRAM FPGAの論理領域や結線用のプログラム素子はSRAM（ルックアップ・テーブル）で構成されているが，ProASIC3/3Eではフラッシュ・メモリで構成されている．また，PLLやメモリ・ブロックを内蔵しており，Xilinx社の「Spartan-II」やAltera社の「Cyclone」と比べて，そん色なく使えそうだ．ただし，乗算器や積和演算器のハード・マクロは組み込まれていない．そのため，高速性やリアルタイム演算が求められる用途には向いていないかもしれない．

型　名	A3P030	A3P060	A3P125	A3P250	A3P400	A3P600	A3P1000
システム・ゲート数	3万	6万	12.5万	25万	40万	60万	100万
タイル数（Dフリップフロップ数）	768	1536	3072	6144	9216	13824	24576
RAM容量	—	18Kビット	36Kビット	36Kビット	54Kビット	108Kビット	144Kビット
フラッシュ・メモリ容量	1Kビット	1Kビット	1Kビット	1Kビット	1Kビット	1Kビット	1Kビット
AES暗号化ISP	非対応	対応	対応	対応	対応	対応	対応
PLL数	—	1	1	1	1	1	1
I/Oバンク数	2	2	2	4	4	4	4

（**a**）ProASIC3

表1
ProASIC3/3Eシリーズの概要
ProASIC3シリーズにはProASIC3とProASIC3Eがある．ProASIC3は小規模用でPLLが1個のみ．ProASIC3EはProASIC3よりも大規模であり，PLL数も6個と多い．他社の競合FPGAとして，Xilinx社のSpartan-Ⅱ/ⅡE，Spartan-3シリーズ，Altera社のCyclone，Cyclone Ⅱシリーズなどが考えられる．ProASIC3Eの最大マスタであるA3PE3000（システム・ゲート換算で約300万ゲート）と対抗するのは，例えばXilinx社のSpartan-3シリーズであればXC3S2000（約200万ゲート）だろう．ただし，Spartan-3はDSPブロックを内蔵しているので，用途によっては同じ土俵で比較することが難しい場合もある．

型　名	A3PE600	A3PE1500	A3PE3000
システム・ゲート数	60万	150万	300万
タイル数（Dフリップフロップ数）	13824	38400	75264
RAM容量	108Kビット	270Kビット	504Kビット
フラッシュ・メモリ容量	1Kビット	1Kビット	1Kビット
AES暗号化ISP	対応	対応	対応
PLL数	6	6	6
I/Oバンク数	8	8	8

（**b**）ProASIC3E

FPGAの場合と異なり，コンフィグレーションROMを必要としません．これによるメリットは，以下の2点です．

1）外付けROMを除いてコスト・ダウン

SRAM FPGAを使用する場合，ボード上に実装するスペースとコンフィグレーションROMの費用を検討しなくてはなりません．ProASIC3/3Eはコンフィグレーション ROMを実装しなくてもよいため，実装スペースの制約が厳しいボードなどに適しているといえます．従来のSRAM FPGA実装品のコスト・ダウン検討時に，「SRAM FPGA＋コンフィグレーションROM」の価格とProASIC3/3E単体の価格を比較してみるとよいかもしれません．ただし，容量や性能など，適切な仕様のデバイスが検討時点で供給されていなければ，うまくコスト・ダウンできないことがあります．

2）プログラミング・データが不法にコピーされない

ボードにコンフィグレーションROMを実装する以上，いくらリード・バック（読み出し）不可などの設定を行っても，コンフィグレーション用の配線を細工することでプログラミング・データを読み取ることができます．これに対してProASIC3/3Eでは基板上にコンフィグレーション配線が存在しないので，細工することは不可能です．SRAM FPGAと比べてセキュリティの面で優位と言えます．

● 基板上の電源配線がシンプルに

近年のSRAM FPGAでは，I/O用電源やコア電源，さらには内部PLL（phase-locked loop）駆動用の電源など，基板上に複数の電源配線を用意しなくてはなりません．これにより，基板の回路設計において，使用する電源ICの選択などにかなりの時間を要しています．一方，ProASIC3/3Eは内部にチャージ・ポンプ回路を内蔵しているため，単一電源による設計が可能となりました．したがって，電源設計において従来のFPGAよりも優位と言えます．

ただし，このチャージ・ポンプ回路の性能はまだよく知られていないので，リプルなどの特性をきちんと把握してから基板の電源設計を行う必要があります．また，チャージ・ポンプ回路をFPGAに組み込んだことにより，発熱についても従来の見積もり方法とは異なると考えられます．パッケージの熱抵抗などの情報を事前に入手しておき，FPGAの発熱がシステム上問題ないかどうかを検討しておかなくてはなりません．

● フラッシュFPGAならではのセキュリティ機能を搭載

ProASIC3/3Eには「FlashLock」と呼ばれるセキュリティ機能があります（**図1**）．この機能はさらに2種類に分けられます．一方はアンチヒューズFPGAと同じように書き込みが一度だけしかできない「永久ロック」，もう一方は128ビットのAES解読かぎを組み込むことにより，認証キーがわからないとプログラミング・データを更新できない

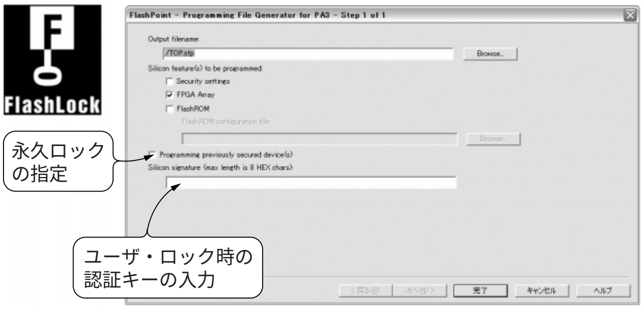


図1　FlashLock設定時のLibero IDEの画面

FlashLock機能には「永久ロック」と「ユーザ・ロック」の2種類がある．図の永久ロックのボックスを有効にすると，2度とプログラミング・データを更新できなくなる．評価ボードなどでこれを有効にして書き込んでしまうと，別の論理回路を実装できなくなるので要注意．ユーザ・ロック時の認証キーのスペースには，HEX（16進数）で8キャラクタ分を入力する．コンフィグレーション時にこの認証キーが一致しないと更新できないしくみになっている．

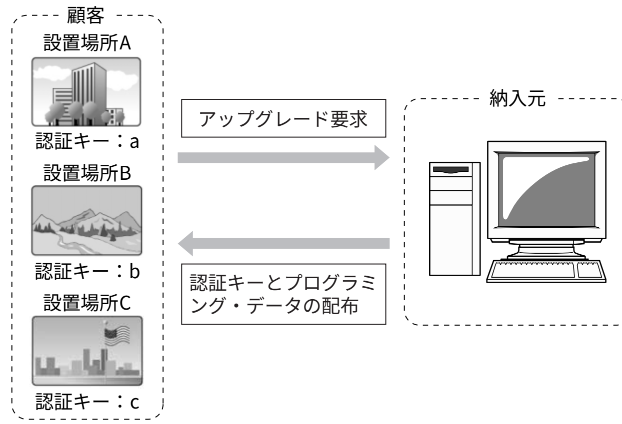


図2　ユーザ・ロックのアプリケーション例

機器の設置場所別に認証キーの異なるプログラミング・データをあらかじめ組み込んでおく．アップグレード要求のあった設置場所に対してのみ，対応する認証キーとプログラミング・データを配布する．これにより，設置場所別のアップグレードが可能になる．

「ユーザ・ロック」です．

永久ロックの機能を有効にしてFPGAにプログラミング・データを書き込むと，その名のとおり，2度とプログラミング・データを更新できなくなってしまいます．つまり，アンチヒューズ式のFPGAと同じ役割を果たせます．今後の回路変更がなく，セキュリティ能力をアンチヒューズFPGAのレベルまで高めたいときは永久ロックを使います．

ユーザ・ロックの場合，プログラミング・データ生成時に8キャラクタ（HEX）の認証キーを埋め込みます．これにより，次のプログラミング・データ更新時に認証キーが一致しないと更新できないしくみになっています．どのようなアプリケーションで利用するかはユーザしだいなのですが，例えば顧客に納入した機器について，その設置場所ごとに異なるアップグレードの要求があったとします．このとき，あらかじめ設置場所ごとに認証キーの異なるプログラミング・データを書き込んでおきます．そして，アップグレードの要求があった設置場所に対してのみ，そのFPGAの認証キーと更新用のプログラミング・データを配布すれば，その設置場所だけの機能更新が可能となります．かりに機能更新用のプログラミング・データを別の設置場所の機器に適用しようとしても，認証キーがわからなければ更新できません（**図2**）．

ちなみにProASIC3/3Eはいかなる場合においてもリード・バック機能は不可です．SRAM FPGAの場合と異なり，ツールによって設定する必要はありません．

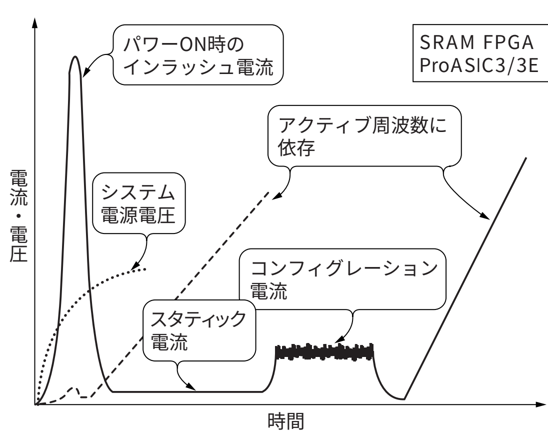


図3　SRAM FPGAとProASIC3/3Eのインラッシュ電流

SRAM FPGAではインラッシュ電流のスパイクを考慮する必要がある．インラッシュ電流のスパイクが原因でシステム全体の電源設計を見直さなければならないケースもある．一方，ProASIC3/3Eではインラッシュ電流のスパイクは発生しない．電源設計時には，ダイナミック動作時の消費電流にだけ注目すればよい．

● インラッシュ電流のスパイクの検討が不要に

5～10年前，FPGAの電源設計においてもっとも気にしなければならない項目はコンフィグレーション時の消費電流でした．しかし，プロセスの微細化によって回路の高速化が進むにつれて，ダイナミック動作時の消費電流がコンフィグレーション時の消費電流を上回るようになりました．それと同時にインラッシュ電流のスパイクが見逃せない問題となっています（**図3**）．

SRAM FPGAはインラッシュ電流のスパイクが大きいのですが，ProASIC3/3Eではこれはほとんど発生しません．インラッシュ電流のスパイクは，電源ICの選択に大きく影響します．

2 フラッシュFPGAの開発の流れ

以下では，ProASIC3/3Eの開発手順や開発ツールについて，「最低，これだけ知っていれば設計できる」というレベルの内容（筆者なりに調べた項目）を紹介します．「これは使いやすいかも…」，「この機能は他社のほうが優れている」といった個人的意見も織り込みながら説明します．

● FPGAの設計フローをおさらい

ここでFPGAの設計フローをおさらいしておきます．なお，ビヘイビア・レベルの工程は含めず，RTL（register transfer level）から設計をスタートすることを前提に説明します．

RTLで用いる言語（HDL）としては，おもにVHDLとVerilog HDLがあります．専用の入力ツールを用いない場合は，ソース・コードをテキスト・エディタなどで直接入力することになります．

HDLのソース・コードが完成したらRTLシミュレーションを行った後，ゲート・レベルのネットリストに変換するために論理合成を行います．論理合成を実行する前に，あらかじめターゲットとなるFPGAを選択する必要があります．生成されたネットリストにはセル情報と結線情報が格納されており，これはFPGAベンダごとにフォーマットが異なります．

FPGAベンダが提供している配置配線ツールを使ってセルの配置とセル間の結線（配線）を行います．このとき，同時にピン配置情報やタイミング情報を付加することによって，システム仕様に合った結果を得ることができます．

配置配線が完了したということは，実際の配線負荷も追加されたことになり，実機動作と同じタイミング情報が得られます．この段階で，実際のタイミング情報を使ってシミュレーションを行います（バックアノテーション・シミュレーション）．これに対して，前述のRTLシミュレーションは，あくまでも論理レベルのシミュレーションです（遅延情報を含まない）．

実際のタイミング情報が付加されると，思わぬところで論理がずれ込み，誤動作を起こすことがあります．そのため，バックアノテーション・シミュレーションで確実に実機相当の検証を行います．

バックアノテーション・シミュレーションが完了したら，最後はチップ全体の性能を検証します．入力ピンのセットアップ/ホールド時間，出力ピンの出力タイミング，内部のフリップフロップ間の最大動作周波数など，システムの条件を満たしているかどうかを確認します．

タイミングに問題がなかったら，コンフィグレーションROMを作成します．シミュレーションとタイミング解析に時間を費やしたROMデータは，実機でもかなり品質の高い動作を示してくれるはずです．めんどうでもシミュレーションはきっちり行いたいものです．

● 開発環境はライセンスの種類が多いので注意

では実際にProASIC3/3Eをターゲットとしたとき，どのようなツールを使えばよいのでしょう．

Actel社製のFPGAを設計する際には「Libero IDE」というツールを使用して設計することになります．Libero IDE

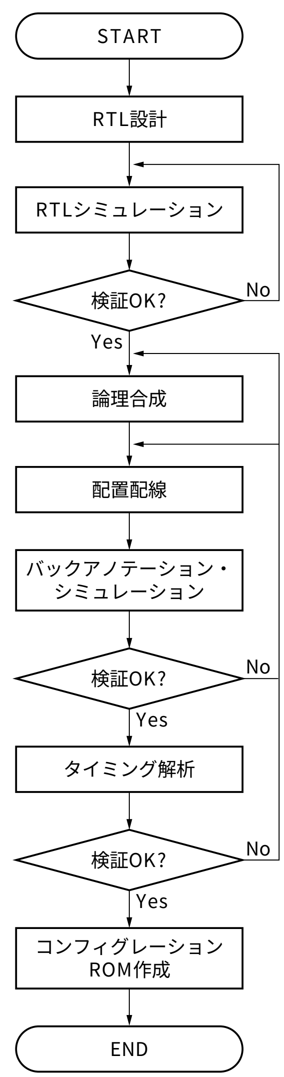


図4 FPGAの設計フロー
RTL設計からコンフィグレーションROM作成までの工程を示す．

3

によって，**図4**の全工程を実施することができますが，筆者はこれまでこのツールを使用したことがありませんでした．

まず，実際に使用する前の調査でわかったことを簡単にまとめておきます．設計データの入力では，ソース・コードを直接入力することもできますし，回路図エディタを利用することもできます．米国Altera社の「MegaWizard」や米国Xilinx社の「CORE Generator」のようなIP（intellectual property）コア生成ツールも用意されています．

GUIを使ってシミュレーション用のテストベンチを作成する「WaveFormer」，およびHDLシミュレータ「Model Sim」も付属しています．シミュレーションのために，あらためてライセンスを取得する必要はありません．論理合成ツールは米国Synplicity社の「Synplify for Actel edition」が付属しています．当然ながら，論理合成ツールはActel社のデバイス限定バージョンです．

配置配線には「Designer」を使用します．Libero IDEで配置配線できるデバイスは，ライセンス形態に依存しています．

「Libero IDE Evaluation」はActel社のすべてのデバイスをサポートしており，無償ライセンスです．ただし，45日間という期間の限定があります．「Libero Gold」では設計できるデバイスに制限があります．無償ライセンスで，ライセンス有効期間は約1年です．これらのツールを利用する際にはActel社のWebサイトで無償ライセンスのライセンス・キーを取得する必要があります．無償とはいえ，ユーザ登録を行い，自分のパソコンでライセンスの環境変数の設定を追加する必要があります．メディアからインストールしてすぐに使えるというわけではなく，お試しで使う場合でもちょっとめんどうな手続きが必要です．また，「Libero Platinum」は有償です．

ちょっとお試しで使ってみたいのであればLibero IDE Evaluationで十分だと思います．一方，お試しといえども時間をかけて評価してみたいのであればLibero Goldがよいかと思います．すでにActel社のFPGAの導入が決まっているのであれば，Libero Platinumを購入してしまったほうがサポートの面から得策だと思います．

● 開発環境をインストールする

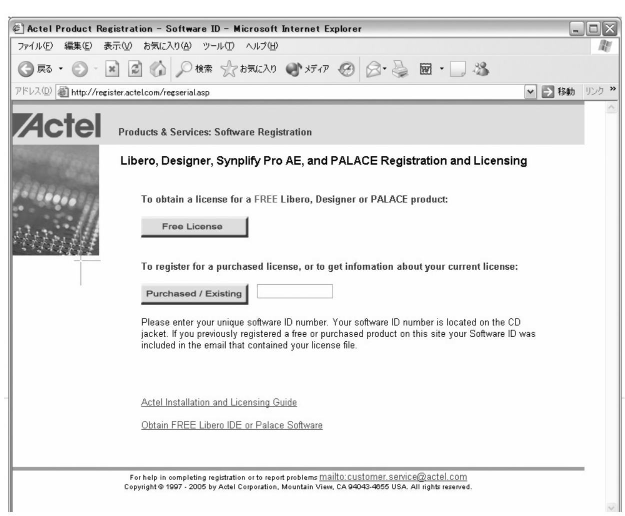
ここでは「Libero IDE Gold 6.2」を使いながら説明します．インストール用のメディアは，今回，Actel社の販売代理店から入手しました[注]．メディアをドライブに挿入すれば，通常は自動的にインストーラが起動します（**図5**）．起動しない場合は，メディア内のsetup.exeを直接実行します．

インストーラ起動画面では，ライセンス形態の選択を行います（今回は「Libero Gold」を選択する）．使用するライセンスを選択し，次へ進むと，ライセンス使用許諾画面になります．使用条件に同意できれば，次に進みます．

Libero Goldが無償で使えると言っても，ライセンス・キーは必要です．インストール完了後，ライセンス・キー取得の確認画面が出てきます．ここではまだライセンス・キーを取得していませんが，後ほどWebからライセンスを取得することになるので，「取得済み」のほうを選んで，いったんウィザードを終了します．



◀図5 Libero IDEインストーラ起動時の画面



▶図6 Liberoの無償ライセンス申請ページ

注：Actel社の販売代理店にはテクセル（http://www.teksel.com/）とジェピコ（http://www.jepico.co.jp/）がある．

● 無償ライセンスを取得する

無償ライセンスでも，Actel社のWebサイトを開いて，ユーザ登録を行わなければなりません．このとき，個人情報を入力する必要があります．申請フォームの安全性が気になる方はActel社に問い合わせ，どのようなセキュリティ機能で個人情報を保護しているのかを説明してもらうとよいと思います．

現在のところ，ライセンス申請に必要な入力フォームはhttp://register.actel.com/RegSerial.aspに用意されています．URLにまちがいがなければ，**図6**のような画面が出るはずです．[Free License] ボタンを選択して次に進んでください．無償ライセンスを選択したら，ライセンスの種別を選択する画面が出てきます．今回は [Libero Gold] ボタンを選択して次に進みます．

ライセンスの種別を選択すると，個人情報や使用言語，パソコンの環境などを入力するフォームが現れます．必要事項はすべて記入しなくてはなりません．入力する必要事項の中に，自分のパソコンのハード・ディスクIDを記入する欄があるので，入力フォームに進む前にハード・ディスクIDをメモしておきましょう．Windowsのコマンドプロンプトを起動し，「vol c:」というコマンドを実行すると，ハード・ディスクIDが表示されます．

入力フォームに記入した事項にまちがいがなければ，[Submit] ボタンを押して送信します（**図7**）．そうすると，Actel社から入力フォームで入力したメール・アドレスへ，license.datというファイルが添付されたメールが送信されてくるはずです．このlicense.datをパソコン上に定義しないとLiberoは起動しません．

license.datをパソコンの任意のフォルダにコピーした後，Windowsのスタート・メニューより「コントロールパネル」を選択し，「システム」を起動してください．「システムのプロパティ」ウィンドウに詳細設定タグがあるのでこれを選択し，ウィンドウ内の [環境変数] ボタンを押します．すると**図8**のような環境変数設定画面が現れます．システム環境変数の変数名を「LM_LICENSE_FILE」，値を「licence.datをコピーした任意フォルダ名 \license.dat」としてください．すでにLM_LICENSE_FILEが設定されている場合は，値のところに追加してください．既存の値の後に；（セミコロン）を追記し，その後に追加する値を入力します．

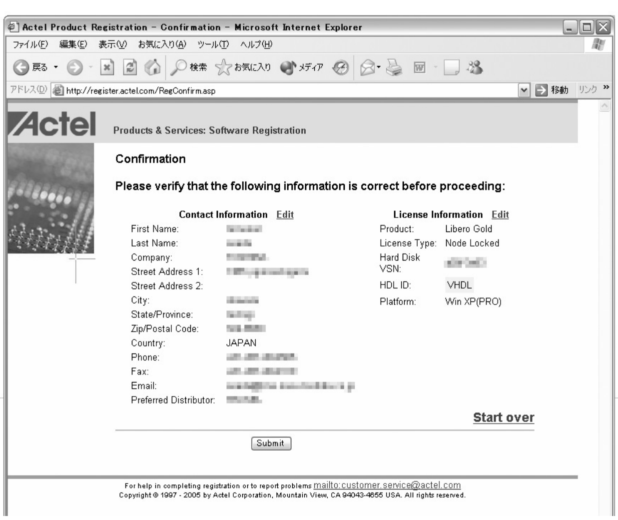
無償とはいえ，Liberoを起動するまでにはそれなりの労力を要します．個人的には，評価用の無償ライセンスであれば，もっと手軽に試用できたほうがありがたいのですが….環境変数までの設定が終われば，いよいよLiberoを起動できますが，ここで一度パソコンを再起動します．

● プロジェクト・ファイルの拡張子の重複に注意

Liberoのインストールが完了すると，デスクトップ上に「Libero IDE v6.2」というアイコンが作成されているはずです．そのアイコンを実行するか，スタート・メニューより「スタート」→「すべてのプログラム」→「Actel Libero IDE v6.2」→「Libero IDE v6.2」を選択してください．

すると**図9**のようなメッセージが出ます．1回目の起動の際には設計プロジェクトが何も設定されていないので，GUIを用いてプロジェクト情報を設定します．

プロジェクト・ファイルの拡張子が「.prj」であるため，SynplifyやSynplify Proのプロジェクト・ファイルの拡張子と重複してしまいます．Windowsの拡張子の設定では，最新の情報を有効とするので，もし既存のSynplify/



◀ **図7 入力フォーム確認ページ**



▶ **図8 環境変数設定ウィンドウ**

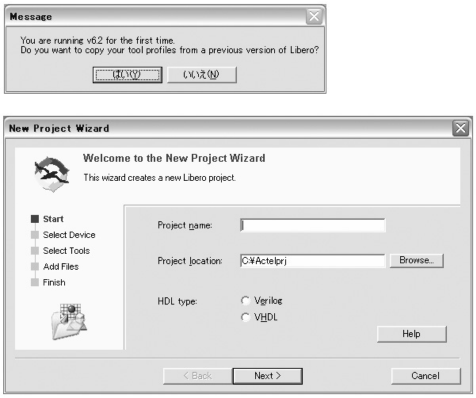
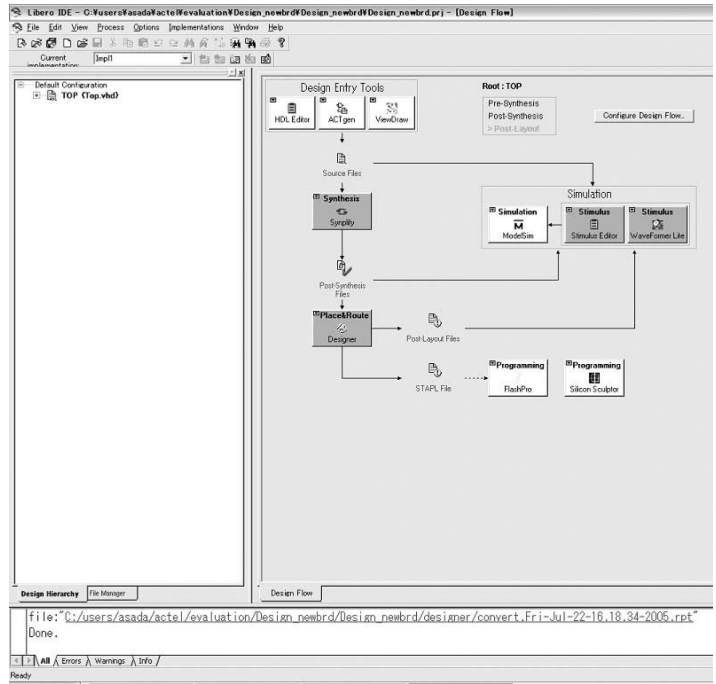


図9　Libero IDEの起動画面



▶ 図10
Libero IDEのメイン画面
設計フローをGUIで確認できる．アイコンをクリックすれば，各ツールが起動するしくみになっているので，初心者でも簡単に扱うことができそうだ．

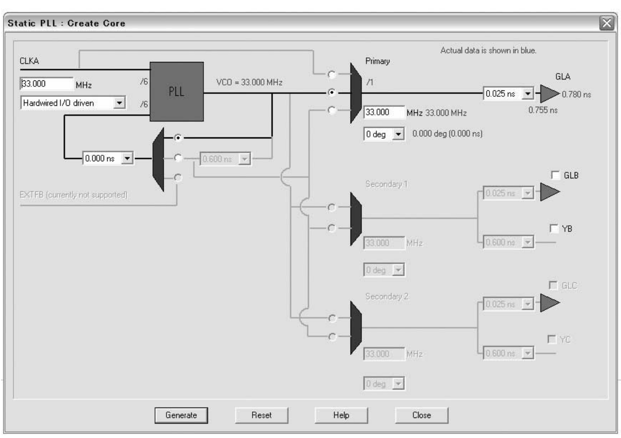


図11　ACTgenのGUI
PLL設定時のGUI．画面で確認しながら入力できる．ソース・コードを直接入力する場合より便利．

Synplify Proのプロジェクトがある場合には，きっとアイコンのマークやファイルの説明がLibero用になっているはずです．Liberoのプロジェクト・ファイルとSynplify/Synplify Proのプロジェクトについてはフォルダを分けるなど，混乱しないように管理する必要があります．

● 設計入力はテキスト，IPコア生成，回路図を用意

Libero IDEのGUIは**図10**のようになっています．この図をもとにLiberoのデザイン・フローを説明します．

Design Entry Toolsでは，3種類の入力方法から好きなものを選択し，RTL設計を行います．エディタを使ってVHDLやVerilog HDLのソース・コードを直接入力する「HDL Editor」，Altera社のMegaWizardやXilinx社のCORE Generatorのように独自のIPコアを生成する「ACTgen」，Actel社のセルを直接呼び出して回路図入力できる「ViewDraw」などがあります．

ACTgenのIPコア生成画面ですが，PLLのような特殊なハード・マクロもACTgenで入力することができます（**図11**）．特殊なハード・マクロについては，ソース・コードで直接コーディングするよりもACTgenを使用したほうが入力が簡単かもしれません．もちろん特殊なIPコアでなくても，簡単なカウンタやデコーダもACTgenを使用して作成できます．

ViewDrawは回路図入力を好む人のために用意されたツールです．1980年代後半～1990年代前半は**図12**のような回路図入力が主流でした．その後，RTLのHDL入力が主流となり，近年ではCベース設計手法が浸透しようとしています．今でも回路図入力をサポートしていることはかなりの驚きですが，これはこれで適材適所に使用すれば効率良く設計できるかもしれません．強くお勧めするわけではありませんが，簡単な論理回路を回路図入力によって組んでみるのも良いかもしれません．

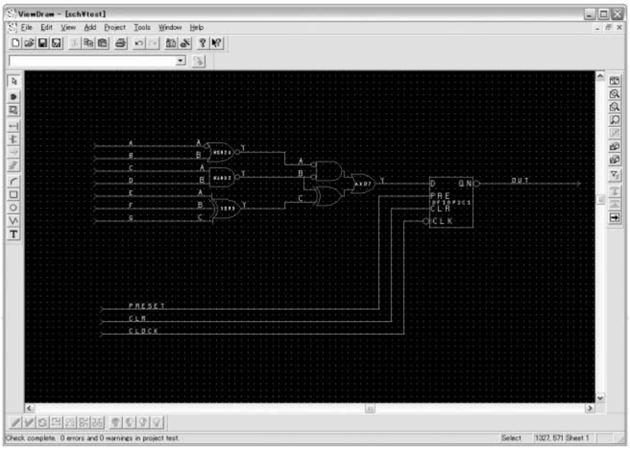


図12　ViewDrawのGUI
このエディタで呼び出すセルはActel社のマクロ・セルである．ゲート・レベル入力なので，すべての設計を回路図で作成すれば論理合成を行う必要はない．

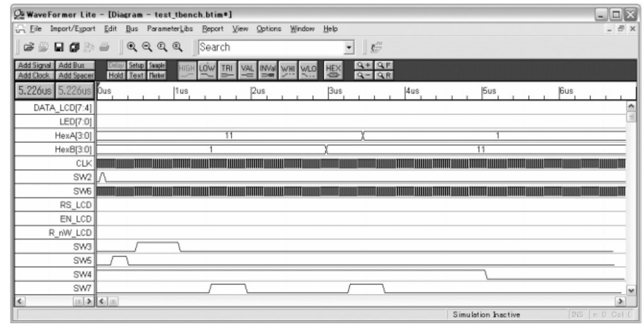


図13　WaveFormerによるテストベンチ作成画面

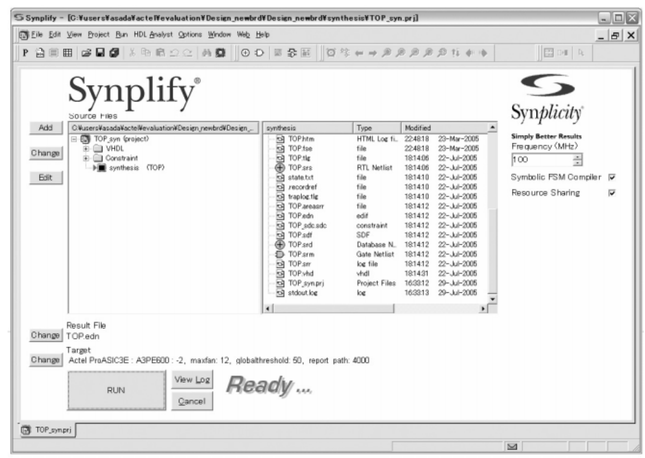


図14　Synplifyの画面

● テストベンチもGUIで生成

設計入力が完了し，コンパイルが完了すれば，設計入力のアイコンが緑色に変化します．こうなると次の工程に進むことができます．次はRTLシミュレーションを行いますが，使用するシミュレータはModelSimです．Liberoのインストール時にActel社専用のModelSimがインストールされます．ModelSimの操作方法は，開発元である米国Mentor Graphics社が提供しているものとなんら変わりがないので，説明を省略します．

シミュレーションを行うには，当然ながらテストベンチが必要となります．Libero IDEでは親切なことにテストベンチ生成もGUIで行えます．

図13のようなWaveFormerというツールを使用することにより，テストベンチを作成できます．筆者もテストベンチについては波形で確認しながら作成できたほうが便利だと考えています．なぜなら，波形なら設計の専門職でない人にでも比較的簡単に説明でき，また，デザイン・レビュー時の資料としても使うことができるからです．ツールを使うことによって設計品質が上がるのであれば，積極的にツールを導入するべきだと思います．

筆者は現在，別のベンダのテストベンチ生成ツールを使用していますが，今回WaveFormerを使った感想としては，初心者でも扱いやすいと感じました．Libero IDEに同梱されているツールの中ではいちばんのお勧めかもしれません．

● 論理合成では最大ファンアウト数などを設定

HDLシミュレータで検証を終えたら，論理合成ツールを使ってゲート・レベルのネットリストを生成します．論理合成には，Libero IDEに同梱されているSynplify for Actel editionを使用します．Synplifyの操作についても，とくにActel editionならではの特別なものはありません．Libero IDEのメイン画面のSynthesisアイコンをクリックすると，Synplifyが起動します(**図14**)．ターゲット・デバイスを選択し，MAX_Fanout(最大ファンアウト数)などのパラメータを設定し，［RUN］ボタンを押します．

正常終了すると，Synthesisアイコンが緑色に変わるので，次に配置配線を行います．

● 配置配線とピン配置情報の入力を行う

Libero IDEのメイン画面のPlace&Routeアイコンを選択すると，**図15**のようなDesignerのメイン画面が現れます．Designerは以下のような機能を含んでいます．

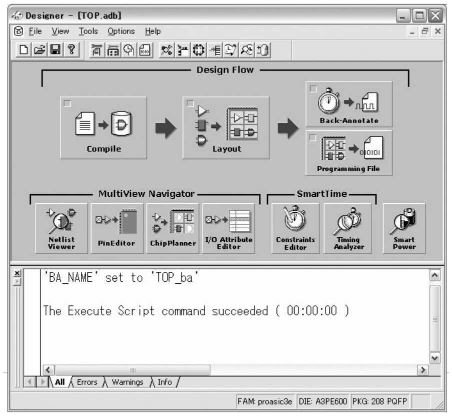
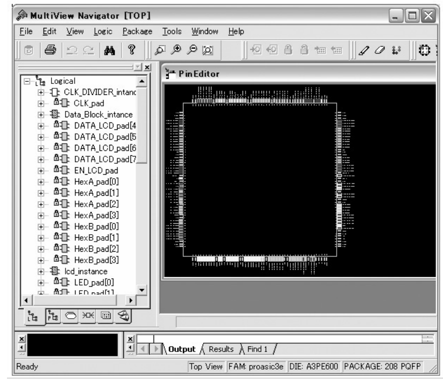
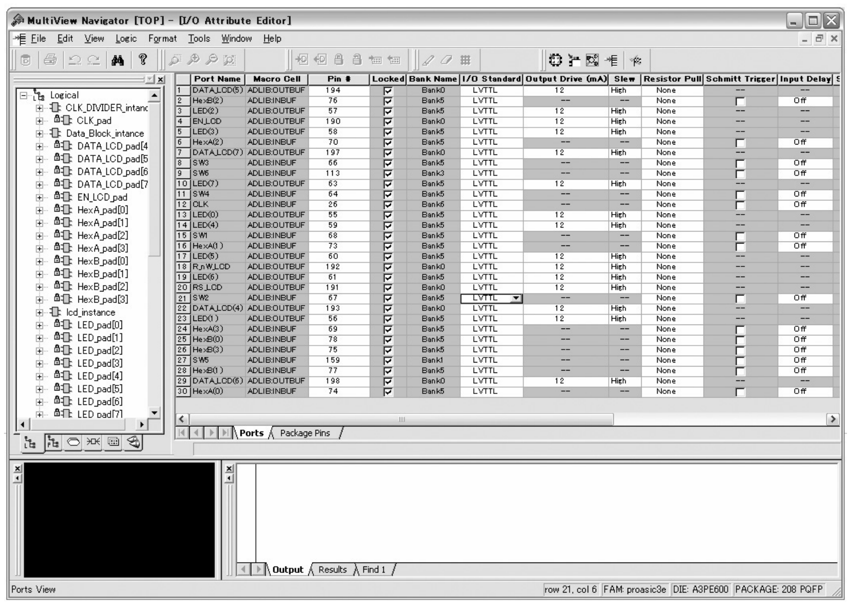


図15　Designerのメイン画面

完了した工程（アイコン）は緑色に変わる．「Programming File」が緑色になれば，プログラミング・ファイル「デザイン名.stp」が作成される．Back-Annotateアイコンを押せば，実配線遅延情報付きのシミュレーション・ソースが出力される．



（**a**）Pin Editor



（**b**）I/O Attribute Editor

図16
ピン配置設定画面とI/Oピンのアトリビュート設定画面

- ゲート・レベル・ネットリストのコンパイル
- ピン配置情報やフロアプランニングの条件入力
- 配置配線
- プログラミング・データの生成
- タイミング解析
- 消費電力の見積もり

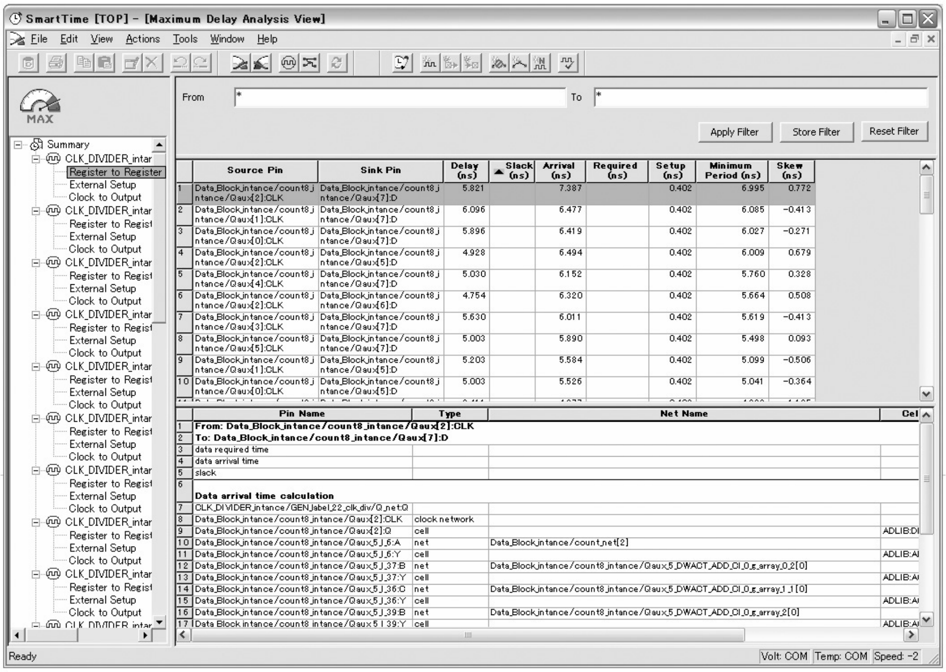
プログラミング・データ生成のアイコンを実行するには，「Compile」と「Layout」が完了している必要があります．ただし，完了していない場合でも，自動的に前工程を実行してくれるので，あまり意識する必要はないようです．

いずれにせよ，配置配線実行前にシステム仕様に基づいてピン配置やI/Oピンの特性を入力しておきます．**図16**に示すPin EditorとI/O Attribute Editorを利用して，配置配線前に必要なパラメータを入力してしまいます．

「Programming File」まで完了すると，アイコンが緑色に変わります．これでプログラミング・データが完成したことになりますが，ここでタイミングを解析したり，消費電力を確認しておく必要があります．タイミング解析にはTimingAnalyzerを，消費電力の確認にはSmartPowerを使用します．このほか，Designerは配置配線後の状態を参照/編集する機能やフロアプランの機能を備えています．これらはFPGAを設計するうえでは必須の機能ではないので，説明を省略します．

● タイミングが目標に達しないときはフロアプランを指定

タイミング解析を行う場合，Designerのメイン画面のTimingAnalyzerアイコンをクリックします．すると，**図17**の画面が現れます．TimingAnalyzer起動時にはすでに



◀ **図17 タイミング解析の画面**

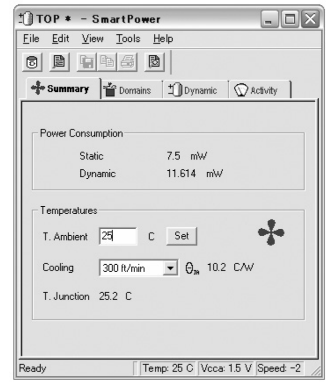


図18 SmartPowerの画面

タイミング計算が行われた状態です．入力のセットアップ/ホールド時間，出力タイミング，内部の動作周波数などをクロックのパス別に確認できます．意図するパスがどの項目にあるのかを探し出すのがちょっとたいへんですが，順を追って探せば，所望の情報が記されているはずです．

タイミング解析の結果がシステム仕様を満たさなかった場合は，再度，配置配線を行う必要があります．しかし，パラメータを何も変えずに配置配線を行っても，改善を得ることは難しいと思います．改善策の一つとしては，フロアプランナの利用があります．クリティカルなパスが最短の配置配線になるように，ユーザの手でフロアプランを指定します．フロアプランナを活用しても改善が得られない場合は，論理そのものを変更しなければなりません．こうなると再びRTL設計に戻ることになり，設計期間にも影響します．そうならないためには，RTL設計の段階でFPGAの配置配線を意識したコーディングを行う必要があります．

● チャージ・ポンプ内蔵なので熱はしっかり解析

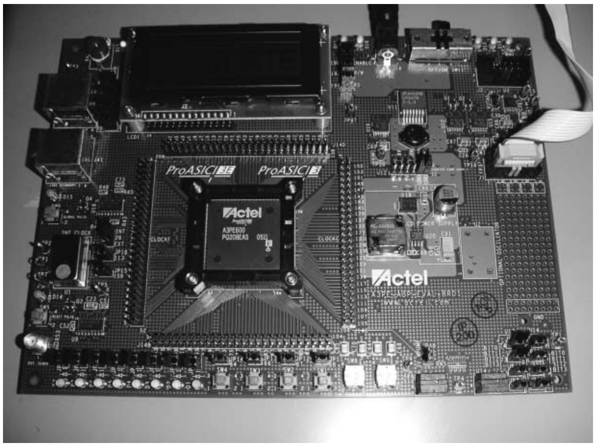
ProASIC3/3EではSRAM FPGAほど消費電力を意識することはないと思いますが，チャージ・ポンプ回路を内蔵していることを考えると，しっかりと熱の解析を行う必要があると思います．

Designerのメイン画面のSmartPowerアイコンをクリックすると，**図18**の画面が現れます．動作周波数などのパラメータを与えることにより，ターゲット・デバイスの消費電力，チップの熱抵抗，チップ周囲の風量を考慮した温度上昇値を算出してくれます．周囲温度(T_a)を加算することにより，チップのジャンクション温度(T_j)を超えていないかどうかをGUI上で確認することができます．

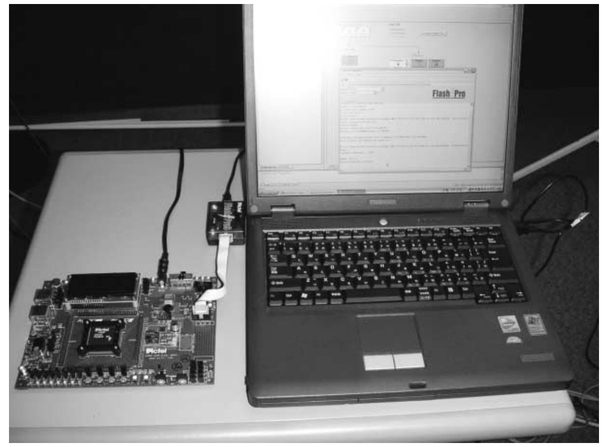
消費電力は動作周波数やゲート規模，スイッチング確率に依存します．一概にProASIC3/3Eならかならず低消費電力になるとは言い切れません．SmartPowerでは，実際の配置配線結果を反映しながら消費電力を算出しています．精度の高い消費電力値であると言えますが，動作周波数などのシステム仕様のパラメータを厳密に入力する必要があります．熱抵抗については，パッケージの大きさによって放熱効果が変わります．例えば同じ設計でも，パッケージをひと回り小さくするだけでも温度上昇率が上がります．FPGAのパッケージの選択も，基板の熱設計を行うときの一つの検討項目であるといえます．

● 評価書き込み時には永久ロックを有効にしない

プログラミング・データをFPGAに書き込む場合，Libero IDEのメイン画面のProgramming FlashProアイコンをクリックします．書き込みに必要なハードウェアで



（a）ProASIC3/3Eの評価ボード



（b）FlashPro3を介してパソコンと接続

写真2　評価ボードにプログラミング・データを書き込む

すが，今回はProASIC3/3Eの評価ボードに同梱されていたFlashPro3を使用しました（**写真2**）．

書き込みの操作方法は，他社のFPGAの書き込み環境と大きな違いはありません．JTAGケーブルの初期化，JTAGチェーン上のデバイスの検出，ターゲット・デバイスとプログラミング・ファイルのリンク，書き込み開始という手順で進めます．

くれぐれも注意していただきたいことは，「Programming previously secured device（s）」を有効にしてプログラミング・データを生成すると，FlashLockの永久ロックが働いてしまうことです．これで書き込みを行うと，FPGAの論理を2度と更新できなくなります．

● ベンダ提供の入力ツールを使うと移植に手間がかかる

Actel社のFPGAをターゲットに設計したことがない筆者にとって，当初は「ProASIC3/3Eって何者？」といった気分でしたが，FPGAベンダや販売代理店の助力もあり，ひとりでRTL設計からプログラミング・データの作成まで行えるようになりました．ツールの操作方法の習得にさほど日数がかからなかったことを考えると，Libero IDEは扱いやすいツールだと思います．FPGA設計のフローを理解されている方なら，すぐに操作に慣れることでしょう．

また，ProASIC3/3Eならではのセキュリティ機能などを有効に活用できれば，これまでにない製品を開発できるのではないかと思います．フラッシュFPGAの価格がSRAM FPGAに対抗できるのであれば，従来FPGAの代替品としても利用できると思います．

ただし，ProASIC3/3Eに限ったことではありませんが，FPGAベンダ専用のRTL設計入力ツールを使ってしまうと，ソースがベンダ専用になってしまい，競合ベンダのFPGAへの移植が困難になるケースがあります．「ほかのベンダのFPGAへの乗り換えは考えていない」という前提であれば問題はないのですが，コスト優先でFPGAを選択する場合は，ほかのFPGAへの移行が必要になるケースもあるかと思います．その際にFPGAベンダ専用のIPコア生成ツールや設計入力ツールを使ってしまうと，そのソースについて再設計が必要となってしまい，2度手間になります．

FPGAベンダ専用のツールは，ほかのベンダのFPGAへ移植する際のネックとなることを忘れてはいけないと思います．ProASIC3/3Eは出荷されて間もないデバイスなので，これから評価される方もいらっしゃるかと思います．本稿がこうしたみなさんのしごとの一助になれば幸いです．

あさだ・とものり
東芝医用システムエンジニアリング（株）

<筆者プロフィール>
浅田朋範．1991年に東芝医用システムエンジニアリングに入社．医用向け画像診断装置のハードウェア設計，FPGAインフラストラクチャの構築業務に従事．入社当初はゲートアレイ，セル・ベースICの開発に従事し，現在は大規模FPGAによるリアルタイム・フィルタ，システム・インターフェース回路の設計を手がけている．